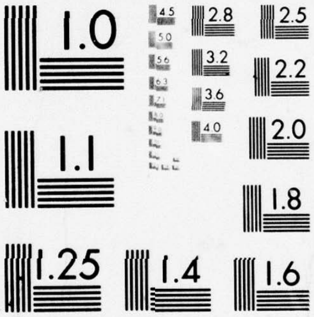
AD-A041 916 GENERAL ELECTRIC CO ARLINGTON VA F/G 5/10
A PRELIMINARY EXPERIMENT TO TEST INFLUENCES ON HUMAN UNDERSTAND--ETC(U)
JUN 77 S B SHEPPARD, L T LOVE N00014-77-C-0158
UNCLASSIFIED TR-77-388100-1 NL

1 OF 1
ADA041 916

END
DATE
FILMED
8-77


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

1
B.S.

TR-77-388100-1

TECHNICAL REPORT

A PRELIMINARY EXPERIMENT TO TEST INFLUENCES ON HUMAN UNDERSTANDING OF SOFTWARE

1 JUNE 1977

GENERAL ELECTRIC
INFORMATION SYSTEMS PROGRAMS
ARLINGTON, VIRGINIA

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER (14) TR-77-3881ØØ-1, 77ISPØØ9 2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) A PRELIMINARY EXPERIMENT TO TEST INFLUENCES ON HUMAN UNDERSTANDING OF SOFTWARE.	5. TYPE OF REPORT & PERIOD COVERED (9) Technical Rept., ~~INTERIM~~ 6. PERFORMING ORG. REPORT NUMBER ~~TIS~~ 77ISP009
7. AUTHOR(s) (10) Sylvia B. Sheppard [redacted] L. Tom Love	8. CONTRACT OR GRANT NUMBER(s) (15) NØØ14-77-C-Ø158 new
9. PERFORMING ORGANIZATION NAME AND ADDRESS General Electric, Information Systems Programs 1755 Jefferson Davis Highway, Suite 200 Arlington, VA 22202	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Work Unit No. NR 197-037
11. CONTROLLING OFFICE NAME AND ADDRESS Department of the Navy Office of Naval Research Engineering Psychology Programs	12. REPORT DATE (11) 1 June [redacted]77 13. NUMBER OF PAGES 19
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) (12) 21p.	15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public use; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

To be published in Proceedings of the Human Factors Society, October 1977, and as a General Electric Technical Information Series Report (77ISP009)

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Understanding — Program Style — Software Science
Control Flow — Mnemonics — Software Psychology
Software Engineering

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Eight experienced programmers were each given three FORTRAN programs to memorize and reproduce functionally, without notes. Three levels of complexity of control flow and three levels of mnemonic variable names were independently manipulated. The experimental design was an incomplete split-plot factorial where each programmer was given one version of each program and all levels of the two primary independent variables. → next page

(Continued)

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

409 446

LB

20. ABSTRACT (Continued)

cont

The participants correctly recalled significantly more statements when the complexity of control flow was reduced. Differences in recall for the three levels of mnemonic variable names were not significant.

A further analysis compared the percent of statements correctly recalled to Halstead's E, a measure of the effort required to code a program. The Pearson correlation coefficient was -0.81, over the 24 data points; thus indicating that Halstead's E is a powerful predictor of one's ability to understand a computer program.

Several changes in the experimental design and the conduct of the experiment itself are recommended for future experimental work in this area.

TR-77-388100-1

TECHNICAL REPORT

A PRELIMINARY EXPERIMENT TO TEST INFLUENCES ON HUMAN UNDERSTANDING OF SOFTWARE

By

S. B. Sheppard and L. T. Love

1 June 1977

Submitted to:

Office of Naval Research
Engineering Psychology Programs
Arlington, Virginia 22217

Contract: N00014-77-C-0158

Work Unit: NR 197-037

GENERAL ELECTRIC COMPANY

INFORMATION SYSTEMS PROGRAMS
1755 Jefferson Davis Highway
Suite 200
Arlington, Virginia 22202

CONTENTS

ILLUSTRATIONS

TABLES

TR-77-388100-1

TECHNICAL REPORT

A PRELIMINARY EXPERIMENT TO TEST INFLUENCES ON HUMAN UNDERSTANDING OF SOFTWARE

Scientific advances in a particular discipline frequently are stimulated by work done in some seemingly unrelated discipline. Recent work in computer science may provide just such an advance to human factors specialists and academic psychologists.

In 1972, Halstead first published his software physics theory (later renamed software science) stating that algorithms have measurable characteristics analogous to physical laws. His objective was to develop quantitative measures of the complexity of computer programs in terms of language level, algorithm purity, programming effort, and programming time. Preliminary tests of the theory have shown very high correlations (greater than 0.90) bwtween the software physics metrics and such dependent measures as the number of bugs in programs (Funami and Halstead, 1975), programming time (Gordon and Halstead, 1975), and quality of programs (Halstead, 1973).

Given the huge individual differences in programming and the inherent complexity and performance variabilities inherent in such tasks, the high correlations suggest that this theory warrants serious review. Our initial experiment was designed to test understanding of computer programs. It asked the following question:

> "Do other independent variables allow us to predict performance more accurately than Halstead's theory?"

This small preliminary experiment has been used to validate materials and procedures for a large experiment to test human understanding of software. The results of the preliminary test were sufficiently interesting to be reported at this time. A larger experiment will be conducted with appropriate modifications as determined by these results.

METHOD

Current literature (Love, 1977; Shneiderman, 1974 and 1977) suggests that the most sensitive measure of whether a person understands a computer program is his ability to memorize the program and reproduce an equivalent program without notes. It would be extremely difficult to reproduce a non-trivial program without some understanding of its function. In this experiment, the participants were asked to reproduce programs and were told that their performance would be evaluated on the functional correctness of the programs written.

Two independent variables, mnemonic names and complexity of control flow, were tested for their relevance to understanding of computer programs. Performance on the programming tasks was also compared to the software physics metrics.

Mnemonic Complexity

Three successively more meaningful levels of mnemonic variable names were defined and tested for each of three programs. M1 had randomly chosen, single-letter variable names. M2 had variable names from 1 to 4 characters in length, selected to be meaningful, but not as mnemonic as M3, the most mnemonic level. For M3, the variable names were chosen to be as meaningful as possible and were 4 to 6 characters in length. (A maximum of 6 characters is allowed by most standard FORTRAN compilers.) No variable had exactly the same name in any two mnemonic versions (Table 1).

Control Flow Complexity

Three levels of complexity of control flow were also defined. S3 was the cleanest control flow, following accepted structured programming rules: no three-way transfer of control statements and no backward transfer of control. S2 and S1 were successively less straightforward. In S2, minor violations of the "top-to-bottom" control flow sequence were allowed. At the S1, or least structured level, there was more frequent use of the GO TO statement and backward transfers of control were allowed. The three-way transfer of control statement [IF ( )-,o,+] was allowed only at the S1 level (Table 2).

Halstead's E

Halstead's theory of software science presents a method for measuring the complexity of software. E, the amount of "effort" required to generate a program, can be calculated from simple counts taken from the program itself. The calculations

Table 1. Mnemonic Variable Names

PROGRAM P1			PROGRAM P2			PROGRAM P3		
M1	M2	M3	M1	M2	M3	M1	M2	M3
D	A	ASCORE	P	X	XVALUE	L	N	NUMBER
E	S	VECTOR	Q	Y	YVALUE	B	X	XVALUE
F	T	TOTAL	R	F	FTABLE	C	Y	YVALUE
B	AV	AVERAGE	M	NC	NUMCOL	E	XCOR	XCOORD
R	SD	STDEV	N	NR	NUMROW	D	PL	PLOT
P	VMIN	VECMIN	V	XB	XBAR	F	YCOR	YCOORD
Q	VMAX	VECMAX	W	YB	YBAR	G	XCO	XCOOR
L	NV	NUMVAR	A	FAC	FACTOR	H	FAC	FACTOR
M	NO	NUMOBS	B	XJI	YCORR1	P	RBOT	RMINI
V	SCNT	COUNT	C	XJ	YCORR2	Q	RTOP	RMAXI
			D	ANS	ANSWER	R	XBOT	XMINI
			O	NO	NUMPTS	S	XTOP	XMAXI
						T	RNG	XRANGE
						U	SCL	XSCALE
						V	CNS	XCONST
						W	STP	STEP
						M	II	IMINNS
						I	IVAL	IVALUE

M1 = RANDOMLY CHOSEN, SINGLE LETTER
M2 = 1 - 4 MEANINGFUL LETTERS
M3 = 4 - 6 MEANINGFUL LETTERS

are based on the two quantities: 1) the sum of the number of distinct operators and operands, and 2) the sum of the total number of operators and operands. From these sums, Halstead derives the number of mental comparisons required to generate a program. (A primary assumption is that human beings use a search technique as efficient as a binary search to retrieve the components of each program instruction.

Table 2. Control Structures Allowed in the Three Levels of Complexity

MOST STRUCTURED (S3)	PARTIALLY STRUCTURED (S2)	LEAST STRUCTURED (S1)
DO (no exits from loops) →	→	→
	DO, exits allowed →	→
IF ( ) assignment →	→	→
IF ( ) GO TO →	→	→
		IF ( ) - , o , +
GO TO (never backward)	GO TO (limited returns above)	GO TO (no constraints)

Since different programming languages produce widely varying numbers of instructions, the number of elementary mental discriminations for each mental comparison varies with the language used. When a correction is made to account for these differences, one can define E in terms of the number of mental discriminations/program:

$$E = \frac{\text{No. of Comparisons}}{\text{Program}} \times \frac{\text{Average No. of Discriminations}}{\text{Comparison}}$$

A thorough discussion of the theory and calculations are beyond the scope of this paper. See Fitzsimmons and Love (1976) or Halstead (1977) for more details.

Testing Procedure

Eight participants were tested in a group. Seven of them had had at least one year of programming experience in FORTRAN. The eighth individual was an experienced COBOL programmer with minimal knowledge of FORTRAN.

Instructions were given to participants orally (Appendix A). Each individual received a program, studied it for 25 minutes, and then attempted to reconstruct it from memory in the next 20 minutes. Three such tasks were given to the participants, with a 20-minute rest period between the last two tasks.

Each individual saw each of the three programs once, each level of mnemonics once, and each level of control flow once. The experimental design (Kirk, 1968;

and Mendenhall, 1968) was a fractional split-plot factorial (Table 3). All participants received a least structured (or most difficult in terms of control flow) program first. Secondly, each received a very structured program. Lastly, each received a program with intermediate structure.

Table 3. Experimental Design for the ONR Pretest

SUBJECTS	ORDER OF OCCURRENCE								
	1			2			3		
1, 4, 7	P1	M3	S1	P2	M2	S3	P3	M1	S2
2, 5, 8	P3	M2	S1	P1	M1	S3	P2	M3	S2
3, 6	P2	M1	S1	P3	M3	S3	P1	M2	S2

P#	=	Program Number
S1	=	Least Structured
S2	=	Moderately Structured
S3	=	Most Structured
M1	=	Least Mnemonic
M2	=	Moderately Mnemonic
M3	=	Most Mnemonic

Informal interviews with the participants following the experiment were conducted to determine their comments on the experimental materials and techniques used.

Programs

Choice of program is a significant factor in any programming test. Previous experience with a particular type of program aids a programmer's understanding greatly. Since it would be difficult to find program functions with which none of the participants had any previous experience, it was decided to use common applications which were readily understandable to most programmers.

The programs chosen are described in Table 4. Each was preceded by a short description of its function (5 to 10 lines). No comments were intersperced in the code, and no indenting of the code was done. All declaration statements appeared at the beginning, and all FORMATs were at the end.

Table 4. Statistics for the Three Programs Tested

PROGRAM #	P1			P2			P3		
PURPOSE OF PROGRAM	CALCULATE MEANS, MAXIMUMS, MINIMUMS AND STANDARD DEVIATIONS FOR A SET OF DATA			PERFORM TWO-DIMENSIONAL INTERPOLATION FOR A POINT, GIVEN A TABLE OF VALUES			SCALE A SET OF COORDINATES (X,Y) AND PRINT A SCATTER PLOT		
STRUCTURE LEVEL	S1	S2	S3	S1	S2	S3	S1	S2	S3
# STATEMENTS IN PROGRAM	46	36	32	56	50	34	83	79	76
# ASSIGNMENT ST.	24	20	16	21	17	14	33	35	34
# CONTROL FLOW ST.	20	14	14	23	22	12	40	34	32
TOTAL # IF STATEMENTS	7	5	5	8	6	3	4	4	3
# GO TO STATEMENTS	7	1	1	8	5	2	9	2	1
HALSTEAD'S E (thousands of mental discriminations)	121	61	59	108	54	40	142	119	80

Different versions of a program varied in length. Programs 1 and 2 averaged 38 and 47 statements respectively, while Program 3, which included a subroutine, averaged 74 statements.

Scoring

All of the programs were scored by the same grader. The criterion for scoring each statement (or each related group of statements) was functional correctness. Variable names and statement numbers different from those in the original program were counted correct when used consistently. Variations of control structures different from the original were noted, but allowed if correct functionally.

The errors in the programs were divided into four categories by statement type: 1) assignment, 2) control flow, 3) other, and 4) extraneous (extra statements the individual added that were unnecessary). Table 5 shows the categorization of statements in the original programs.

The individual errors within statements were also counted, and they fell into three general categories: syntax errors, assignment errors, and transfer of control errors.

Table 5. Classification of Statement Type

ASSIGNMENT	TRANSFER OF CONTROL	OTHER
=	DO	SUBROUTINE
READ	IF ( ) -, o, +	FORMAT
WRITE	IF ( ) T, F	END
DIMENSION	IF ( ) TO GO	
INTEGER	IF ( ) ASSIGNMENT	
DATA	CONTINUE	
COMMON	RETURN	
	STOP	
	CALL	
	GO TO	

RESULTS

There were not enough errors to do an analysis by error type. Therefore, the analyses concentrated on four dependent variables, relating to statements only:

PERCENT OF STATEMENTS ATTEMPTED
PERCENT OF STATEMENTS CORRECT
PERCENT OF ASSIGNMENT STATEMENTS CORRECT
PERCENT OF CONTROL FLOW STATEMENTS CORRECT

The overall mean of the percentage of statements done correctly was 50.3, indicating that the difficulty of the task assigned was neither too difficult nor trivial.

Analyses of Variance

Analyses of variance for each of the dependent variables (Nie, 1975) showed no significant effects caused by mnemonic level, although it appeared that there might be a trend to do slightly better with the single-letter, randomly assigned variable names (Figure 1). This is not the result one would expect here, since longer, meaningful names are intuitively easier to understand. Further work might investigate more levels of meaningfulness to try to explain this phenomenon.

Clearly, a more straightforward control path should produce a more understandable program. As expected, significant differences were produced among levels of control flow for all four dependent variables.

This result must be viewed with some retraint, however, because for each program the total number of statements decreased as the control flow became easier, thus making it possible to complete a greater percentage of the total statements while not actually completing many more statements. Further, one participant did not attempt to complete any statements on the least structured version of the longest program, P3. The percent of statements correct on the other two programs was 74 and 81, indicating that Control Flow Level 1 values in Figure 1 would have been increased if the individual had completed any statements correctly.

One particularly interesting result (Figure 1) is that the percentages of control flow statements done correctly are markedly less than the percentages of assignment statements done correctly in each case. This suggests that control flow statements are more difficult to reproduce than assignment statements, for some as yet unknown reason.

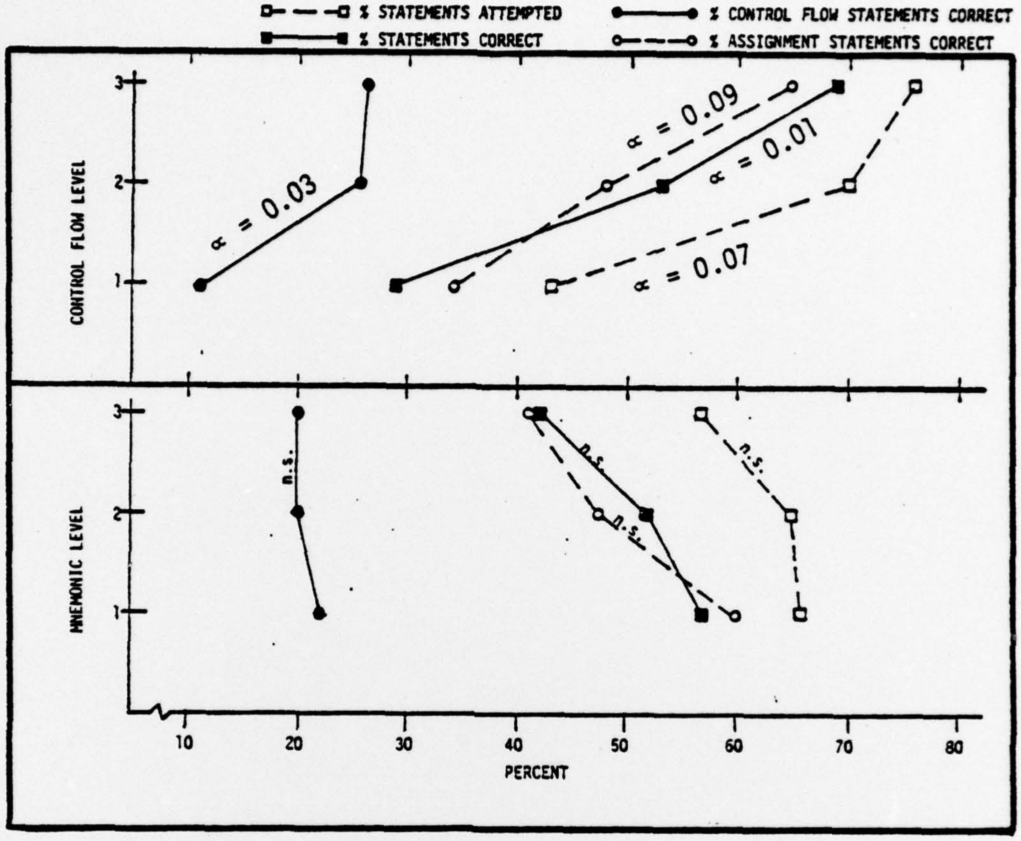


Figure 1. Analysis of Variance Results for Three Mnemonic Levels and Three Levels of Control Flow

Regression Analysis

A multiple regression analysis (Kerlinger, 1973) was done, using effect coding for the independent categorical variables, participants, program, level of control, and level of mnemonic variable names. The particular split-plot factorial design used in this experiment had the order of presentation confounded with level of control flow, making it impossible to assess these variables separately. The experiment was designed under the false assumption that affects due to control flow and order of presentation were monotonic and linear. These affects will be separated in future experiments.

Table 6 presents the change in R Square due to the independent variables and the significance of the regression equation as each variable was included. A total of 92 percent of the variance accounted for is extremely respectable.

Table 6. Results of Multiple Step-Wise Regression Analysis

ORDER OF ENTRY INTO REGRESSION EQUATION	VARIABLE	PERCENT CHANGE IN R SQUARE	SIGNIFICANCE OF REGRESSION EQUATION
1	PROGRAM	43	0.003
2	SUBJECT	18	0.065
3	CONTROL FLOW/ ORDER OF PRESENTATION	26	0.001
3	MNEMONIC LEVEL	5	0.001
4	INTERACTION CXM	0	*
TOTAL		92	

Programs accounted for the largest amount of change in R Square (43 percent). Comparing the total number of statements per program to the percentage of statements done correctly, the correlation coefficient was found to be -0.70. Program P3, which averaged 74 statements, as opposed to 38 and 47 for P1 and P2 respectively, largely accounted for the decrease in the number of statements correctly done.

For each of the programs used, a measure of psychological complexity was computed, using a current theory of software science. A measure of the mental effort required to create a program, described by Professor Halstead of Purdue University (Halstead, 1975), had previously been shown to be an excellent predictor of 1) the number of errors in a program, and 2) the time required to implement a program.

Halstead's E for each of the programs used is shown in Table 4. It was found to be correlated -0.81 with the precentage of statements correctly recalled (Figure 2). Entering only E into the regression equation (Table 7), it accounted for 65 percent of the variance.

* This interaction could not be evaluated with the experimental design used and was, therefore, not entered into the regression equation.

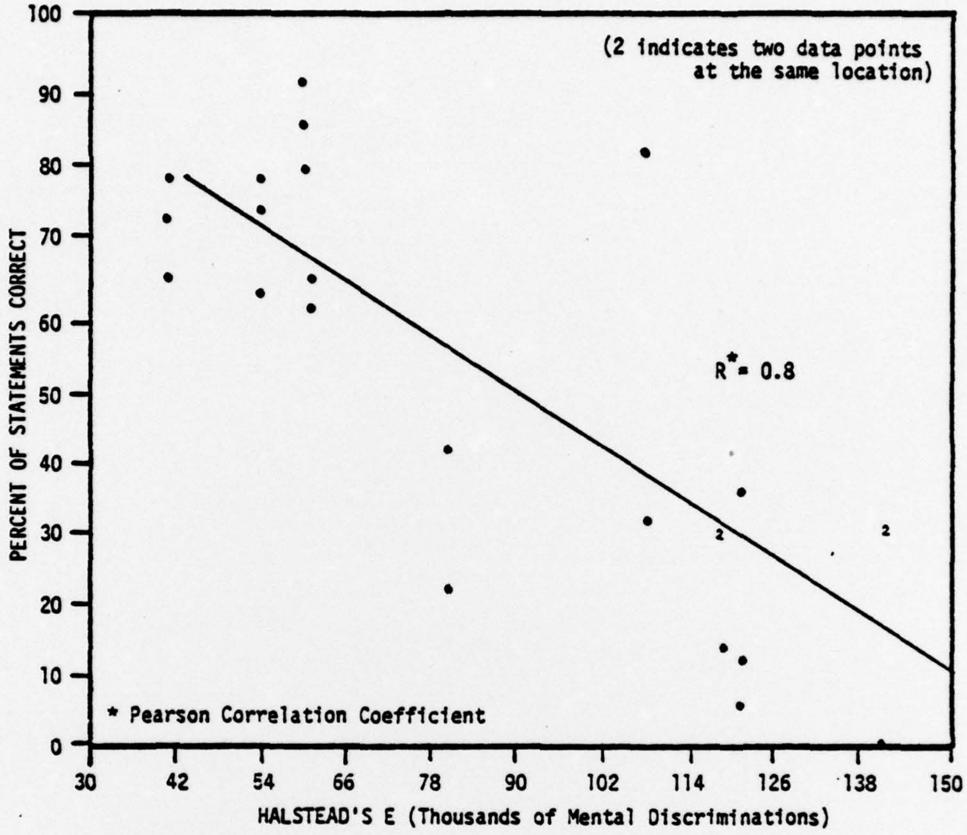


Figure 2. A Comparison of Halstead's E and the Percent of Statements Correctly Completed (R = 0.8) for Each Program

Table 7. Results of Multiple Step-Wise Regression Analysis with Halstead's E

ORDER OF ENTRY INTO REGRESSION EQUATION	VARIABLE	PERCENT CHANGE IN R SQUARE	SIGNIFICANCE OF REGRESSION EQUATION
1	HALSTEAD'S E	65	0.001
2	PROGRAM	3	0.001
3	PARTICIPANT	15	0.001
4	MNEMONICS (M)	9	0.001
4	CONTROL FLOW (C)/ORDER	0	0.001
5	INTERACTION C X M	0	*
TOTAL		92	

* This interaction could not be evaluated with the experimental design used and was, therefore, not entered into the regression equation.

Comparing Tables 6 and 7, we can recognize the power of Halstead's E as a predictor. In Table 6, programs and control flow accounted for 69 percent of the variance. In Table 7, Halstead's E, programs, and control flow accounted for 68 percent of the variance with programs and control flow responsible for only 3 percent. Clearly, we would improve future experiments with a more uniform distribution of the E factor. The distribution in Figure 2 appears to be skewed and bimodal, whereas in future experiments the range of program materials will permit E to assume a more continuous distribution.

CONCLUSIONS

Only 24 observations were obtained in this incomplete, split-plot factorial design, and it is not appropriate to draw far-reaching conclusions on the basis of this work. However, several findings do merit discussion.

When evaluating levels of mnemonic variable names, we expected longer, more meaningful names to make programs easier to understand. However, no significant differences were found among the levels of variable names. Contrary to intuition, the single, randomly chosen letters resulted in better (albeit nonsignificant) recall than either 1 to 4 or 4 to 6 "meaningful" letters. Further work must evaluate variable names more comprehensively.

Control flow made a large difference in performance, as expected. The more structured programs were recalled significantly better than the less structured ones. In addition, the percentage of control flow statememts correctly completed across all conditions was dramatically less than the percentage of assignment statements correctly completed, indicating that control flow statements themselves are more difficult to understand.

Halstead's E, a measure of the effort required to complete a program, was found to be highly correlated (-0.81) with the percent of statements correctly completed. A correlation of this magnitude is impressive in an experiment designed to test counterintuitive theoretical predictions.

We recognize that the effects of the independent variables, complexity of control flow, and mnemonic variable names cannot be unambiguously determined as a result of this experiment. Control flow appears to be highly significant, yet possible effects due to the confounding with order of presentation cannot be ruled out. Further work is needed to determine the effects of variable names in a more sensitive study.

REFERENCES

Fitzsimmons, A., and Love, L.T. A Review and Critique of Halstead's Theory of Software Physics (TIS No. 76ISP004). Arlington, VA: General Electric Company, December 1976.

Funami, U., and Halstead, M.H. A Software Physics Analysis of Akiyama's Debugging Data (Tech. Rep. 1944). Layfayette, IN: Computer Science Department, Purdue University, 1975.

Gordon, R.D., and Halstead, M.H. An Experiment Comparing Fortran Programming Times with the Software Physics Hypothesis (Tech. Rep. 167). Layfayette, IN: Computer Science Department, Purdue University, 1975.

Halstead, M.H. An Experimental Determination of the "Purity" of a Trivial Algorithm (Tech. Rep. 73). Layfayette, IN: Computer Science Department, Purdue University, 1973.

Halstead, M.H. Elements of Software Science. New York: Elsevier North-Holland Inc., 1977

Kerlinger, F.N., and Pedhazur, E.J. Multiple Regression in Behavioral Research. New York: Holt, Rinehart and Winston, 1973.

Kirk, R.E. Experimental Design: Procedures for the Behavioral Sciences. Belmont, CA: Wadsworth, 1968.

Love, L.T. Relating Individual Differences in Computer Programming Performance to Human Information Processing Abilities. Unpublished doctoral dissertation, University of Washington, 1977.

Mendenhall, W. Introduction to Linear Models and the Design and Analysis of Experiments. Belmont, CA: Wadsworth, 1968.

Nie, N.H. SPSS: Statistical Package for the Social Sciences (2nd ed.). New York: McGraw-Hill, 1975.

Shneiderman, B. Two Experiments in Programming Behavior (Tech. Rep. 17). Bloomington, IN: Computer Science Department, University of Indiana, 1974.

Shneiderman, B. Measuring Computer Program Quality and Comprehension (Tech. Rep. 16). College Park, MD: Department of Information Systems Management, University of Maryland, 1977.

OFFICE OF NAVAL RESEARCH, CODE 455
TECHNICAL REPORTS DISTRIBUTION LIST

Director, Engineering Psychology
Programs, Code 455
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Dr. Robert Young
Director, Cybernetics Technology Office
Advanced Research Projects Agency
1400 Wilson Blvd.
Arlington, VA 22209

Office of Naval Research
International Programs
Code 1021P
800 North Quincy Street
Arlington, VA 22217

Director, Information System
Program, Code 437
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Commanding Officer
ONR Branch Office
ATTN: Dr. Charles Davis
536 South Clark Street
Chicago, IL 60605

Commanding Officer
ONR Branch Office
ATTN: Mr. R. Lawson
1030 East Green Street
Pasadena, CA 91106

Director, Naval Research Laboratory
Technical Information Division
Code 2627
Washington,D.C. 20375

Office of the Chief of Naval
Operations, OP987P10
Personnel Logistics Plans
Department of the Navy
Washington, D.C. 20350

Defense Documentation Center
Cameron Station
Alexandria, VA 22314

Col. Henry L. Taylor, USAF
OAD(E&LS) ODDR&E
Pentagon, Room 3D129
Washington, D.C. 20301

Director, Statistics and
Probability Program, Code 436
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Commanding Officer
ONR BranchOffice
ATTN: Dr. J. Lester
495 Summer Street
Boston, MA 02210

Commanding Officer
ONR Branch Office
ATTN: Dr. E. Gloye
1030 East Green Street
Pasadena, CA 91106

Dr. M. Bertin
Office of Naval Research
Scientific Liaison Group
American Embassy, Room A-407
APO San Francisco, CA 96503

Mr. John Hill
Naval Research Laboratory
Code 5707.40
Washington, D.C. 20375

Mr. Arnold Rubinstein
Naval Material Command
NAVMAT 0344
Department of the Navy
Washington, D.C. 20360

Commander
Naval Air Systems Command
Human Factors Programs, AIR 340F
Washington, D.C. 20361

Mr. T. Momiyama
Naval Air Systems Command
Advance Concepts Division, AIR 03P34
Washington, D. C. 20361

Mr. James Jenkins
Naval Sea, Systems Command
Code 06H1-3
Washington, D.C. 20362

Director
Behavioral Sciences Department
Naval Medical Research Institute
Bethesda, MD 20014

Chief, Aerospace Psychology Division
Naval Aerospace Medical Institute
Pensacola, FL 32512

Navy Personnel Research and
Development Center
Management Support Department
Code 210
San Diego, CA 92152

Mr. A.V. Anderson
Navy Personnel Research and
Development Center
Code 302
San Diego, CA 92152

LCDR William Moroney
Human Factors Engineering Branch
Code 1226
Pacific Missile Test Center
Point Mugu, CA 93042

Human Factors Division
Naval Ocean Systems Center
Department of the Navy
San Diego, CA 92152

Commander
Naval Air Systems Command
Crew Station Design, AIR 5313
Washington, D. C. 20361

Commander
Naval Electronics Systems Command
Human Factors Engineering Branch
Code 4701
Washington, D.C. 20360

Dr. James Curtin
Naval Sea, Systems, Command
Personnel & Training Anaylses Office
NAVSEA 074C1
Washington, D.C. 20362

Dr. George Moeller
Human Factors Engineering Branch
Submarine Medical Research Laboratory
Naval Submarine Base
Groton, CT 06340

Bureau of Naval Personnel
Special Assistant for Research Liaison
PERS-OR
Washington, D.C. 20370

Dr. Fred Muckler
Navy Personnel Research and
Development Center
Manned Systems Design, Code 311
San Diego, CA 92152

LCDR P.M. Curran
Human Factors Engineering Branch
Crew Systems Department
Naval Air Development Center
Johnsville
Warminster, PA 18974

Human Factors Section
Systems Engineering Test Directorate
U.S. Naval Air Test Center
Patuxent River, MD 20670

Human Factors Engineering Branch
Naval Ship Research and Development
Center, Annapolis Division
Annapolis, MD 21402

Dr. Jerry Lamb
Display Branch
Code TD112
Naval Underwater Systems Center
New London, CT 06320

Human Factors Department
Code N215
Naval Training Equipment Center
Orlando, FL 32813

Dr. Gary Poock
Operations Research Department
Naval Postgraduate School
Monterey, CA 93940

Mr. J. Barber
Headquarters, Department of the Army, DAPE-PBR
Washington, D. C. 20546

Dr. Edgar M. Johnson
Organization and System Research Laboratory
U.S. Army Research Lab.
1300 Wilson Blvd.
Arlington, VA 22209

U.S. Air Force Office of Scientific Research
Life Sciences Directorate, NL
Bolling Air Force Base
Washington, D.C. 20332

Lt. Col. Joseph A. Birt
Human Engineering Division
Aerospace Medical Research Laboratory
Wright Patterson AFG, OH 45433

Dr. Robert Williges
Human Factors Laboratory
Virginia Polytechnic Institute
130 Whittemore Hall
Blacksburg, VA 24061

Naval Training Equipment Center
ATTN: Technical Library
Orlando, FL 32813

Dr. Alfred F. Smode
Training Analysis and Evaluation Group
Naval Training Equipment Center
Code N-OOT
Orlando, FL 32813

Dr. A.L. Slafkosky
Scientific Advisor
Commandant of the Marine Corps
Code RD-1
Washington, D.C. 20380

Dr. Joseph Zeidner
Director, Organization and Systems Research Laboratory
U.S. Army Research Institute
1300 Wilson Boulevard
Arlington, VA 22209

Technical Director
U.S. Army Human Engineering Labs
Aberdeen Proving Ground
Aberdeen, MD 21005

Dr. Donald A. Topmiller
Chief, Systems Engineering Branch
Human Engineering Division
USAF AMRL/HES
Wright-Patterson AFB, OD 45433

Air University Library
Maxwell AFB, AL 36112

Dr. Arthur I. Siegel
Applied Psychological Services, Inc.
404 East Lancaster Street
Wayne, PA 19087

Dr. Gershon Weltman
Perceptronics, Inc.
6271 Variel Avenue
Woodland Hills, CA 91364

Dr. H. Rudy Ramsey
Science Applications, Inc.
40 Denver Technological Center West
7935 East Prentice Avenue
Englewood, CO 80110

Dr. Robert G. Pachella
University of Michigan
Department of Psychology
Human Performance Center
330 Packard Road
Ann Arbor, MI 48104

Dr. Stanley Deutsch
Office of Life Sciences
HQS, NASA
600 Independence Avenue
Washington, D.C. 20546

Journal Supplement Abstract Service
American Psychological Association
1200 17th Street, N.W.
Washington, D.C. 20036

Director, Human Factors Wing
Defence & Civil Institute
Environmental Medicine
Post Office Box 2000
Downsville, Toronto, Ontario
CANADA

Dr. J.A. Swets
Bolt, Beranek, and Newman, Inc.
50 Moulton Street
Cambridge, MA 02138

Dr. L.A. Miller
IBM
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Dr. Jesse Orlansky
Institute for Defense Analyses
400 Army-Navy Drive
Arlington, VA 22202

Director, National Security Agency
ATTN: Dr. Douglas Cope
Code R51
Ft. George G. Meade, MD 20755

Dr. William A. McClelland
Human Resources Research Office
300 N, Washington Street
Alexandria, VA 22314

Dr. A.D. Baddeley
Director, Applied Psychology Unit
Medical Research Council
15 Chaucer Road
Cambridge, CB2 2EF
ENGLAND

APPENDIX A
SCRIPT FOR TESTING

Hi!

I'm ________________________, and I'm with General Electric's Information Systems Programs in Arlington, Virginia. Our group is interested in experimenting with different types of programs to see how we can make a programmer's job easier. Our work is being funded by the Office of Naval Research, and we are paying for your services from that contract.

Let me assure you that we are evaluating alternative programming techniques and *not* evaluating programmers. The experiment was designed so that you serve as your own control. Your performance on a program will be compared only to *your* performance on other programs. Your only competition is yourself. As a result, it is not necessary for you to put your names on your papers.

If you look over at your neighbor during the test, you will see that each of you is doing a different chore. If someone else finishes earlier than you, don't be concerned about it. He or she will have been working on something else, which might not take as much time as your chore.

We are going to begin this morning by asking you to study a FORTRAN program for 25 minutes. You may write anything, draw flow charts, or whatever helps you to understand the program. When the 25 minutes are up, I'll ask you to hand in the programs and all of your notes. We'll then give you 20 minutes to reconstruct the program from memory as closely as possible to the original.

On some of these programs, you may think of a better way to rewrite it, but I'm asking you to restrain your creativity and just try to reproduce the original program as accurately as possible, line by line. That means using the same constructions. If we have a subroutine, please use a subroutine. If we have a DO loop, please do it with a DO loop. However, it's not necessary to memorize statement numbers. If your statements are numbered differently from mine, it's O.K., as long as the program still does the same job. If you can use the same variable names, that will help me to score the programs, but it will be O.K. if you can't do that.

Does anyone have any questions?